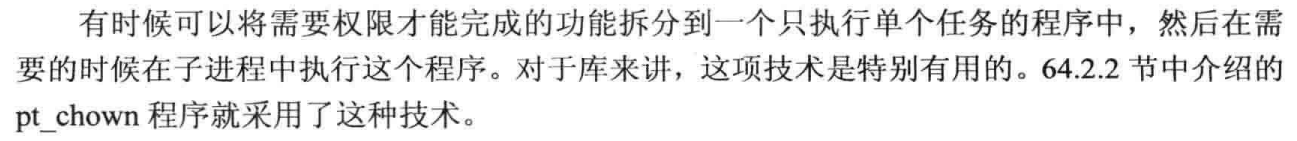
第三十八章

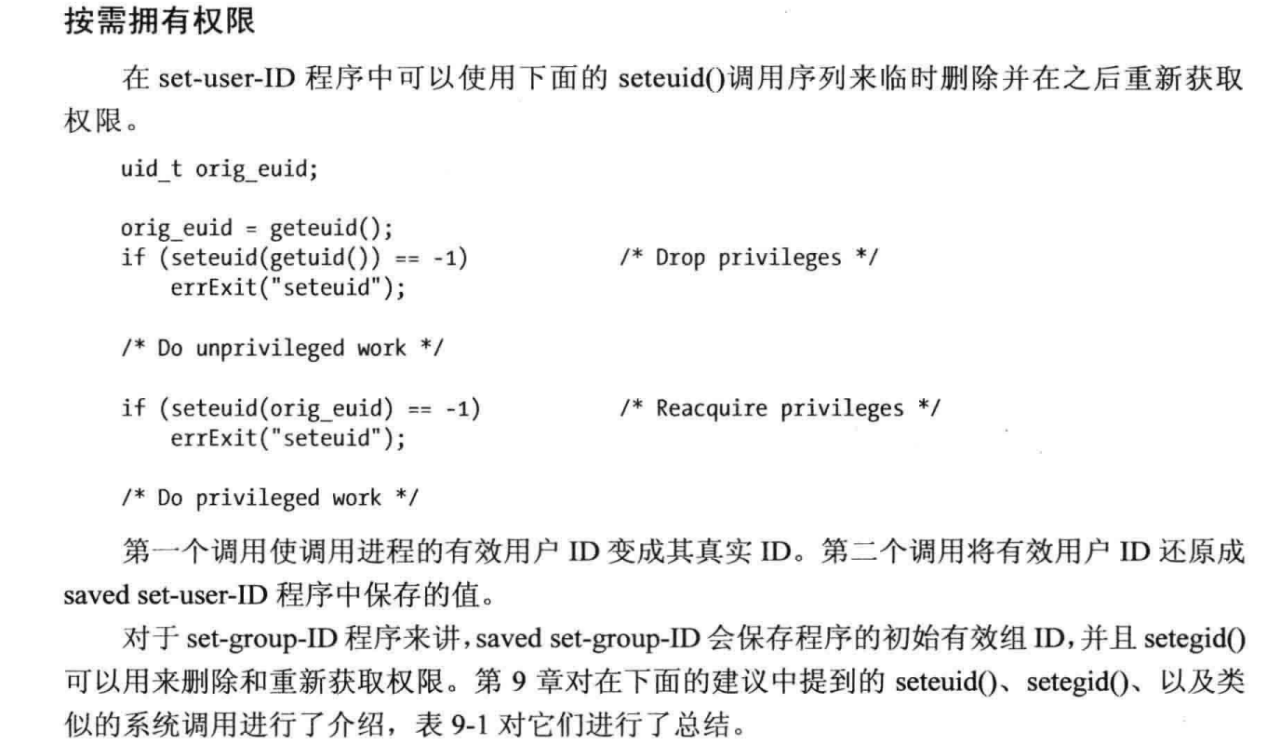
1. 编写安全的特权程序：
2. 是否需要一个set-user-id或set-group-id程序：
3. 尽量避免编写这种程序。
4. 遇到一个问题：



1. 按需分配身份。
2. 对于需要修改文件的情况，可以利用文件组解决。
3. 以最小权限操作：

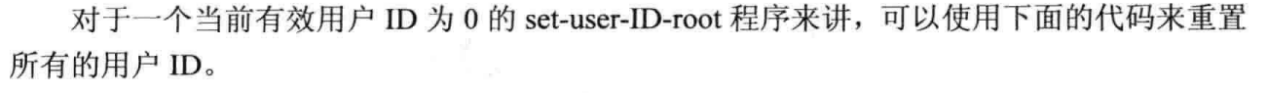
在执行其他不需要特权工作的时候，应该禁用这些权限。如果之后再也不用请求这些权限，应该永久删除。Saved set-user-id就是为这些设计的。

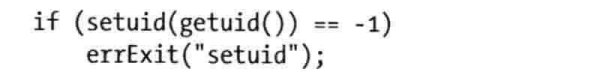
1. 按需拥有权限：

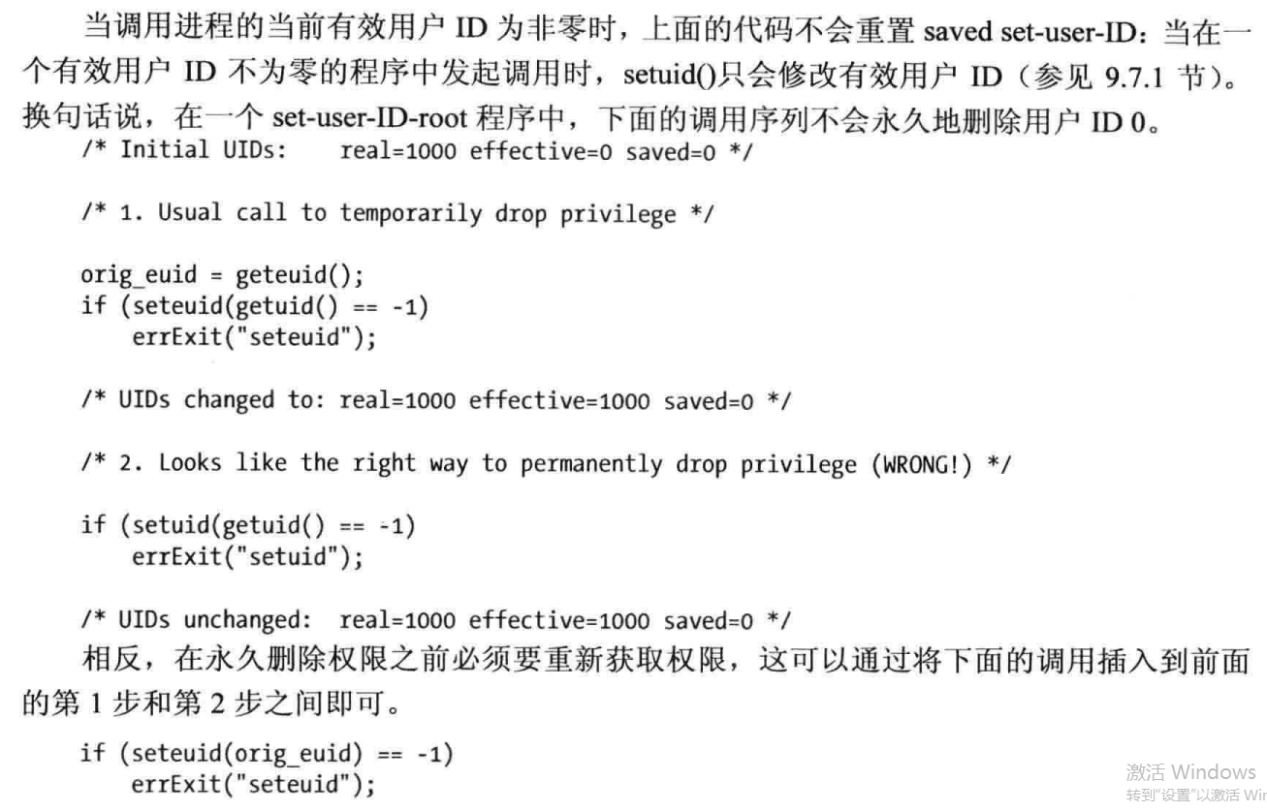


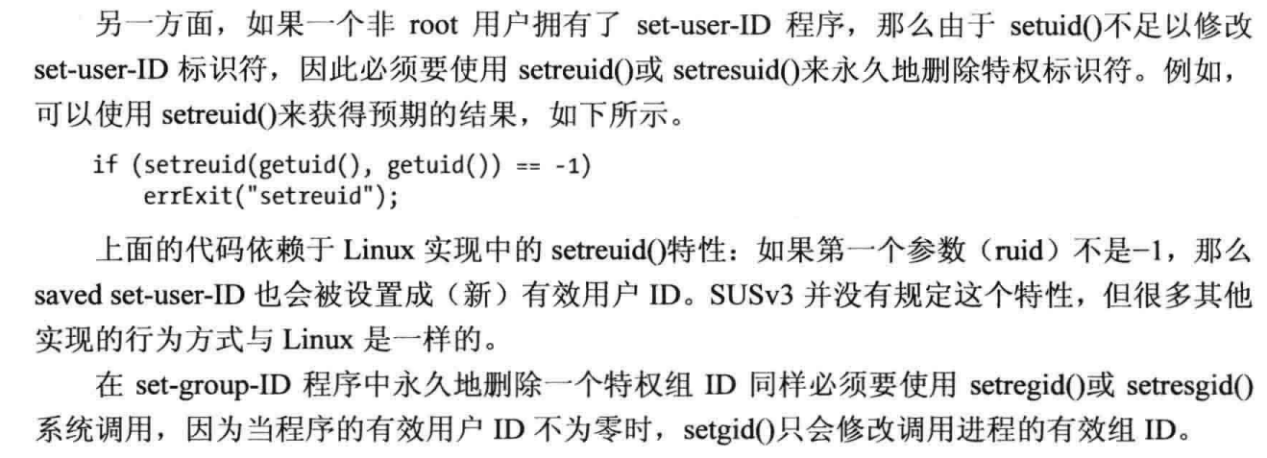
1. 永久删除权限：

是通过将用户所有ID重置为真实用户ID实现的。Saved set-user-id初始值是跟有效ID一样的。





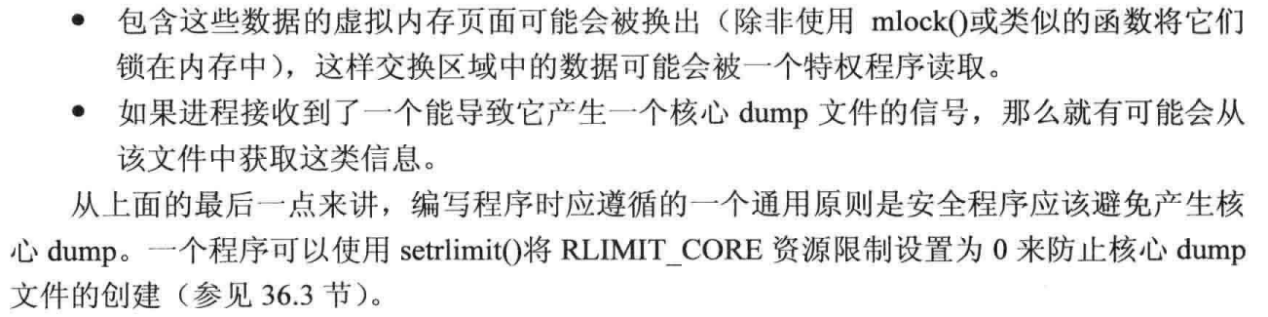




上述修改进程身份需要注意一下信息：

1. 修改进程身份的调用在不同进程和不同权限的表现是不一样的。
2. 在Linux上与能力也有关系。
3. 修改完后要重新验证。
4. 有效ID是决定操作的权力。
5. 小心执行程序：
6. 在执行另一个程序之前永久地删除权限：注意，setuid(getuid)在大多数情况下是可用的。因为，exec调用成功后，会把有效ID复制到saved-set-user-id里面。
7. 避免执行一个拥有权限的shell。
8. 在exec之前关闭所有用不到的文件描述符。
9. 避免暴露敏感信息：

在程序中要及时删除内存中的敏感信息，原因如下：

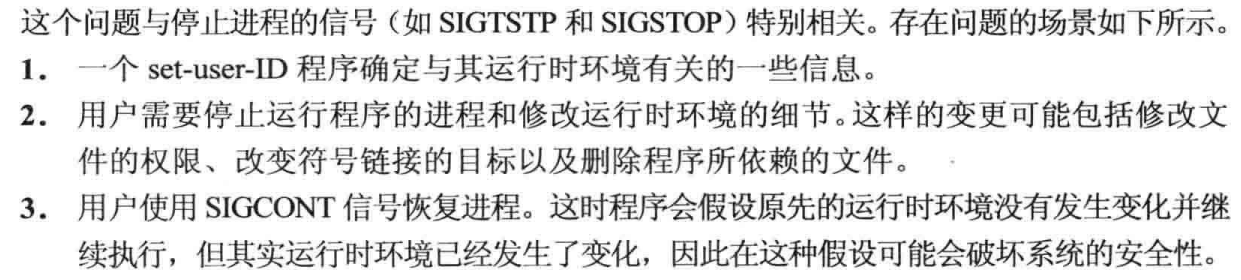


1. 确定进程边界：

控制发生安全问题时造成损失的办法：

1. 使用能力。
2. 使用chroot监牢。
3. 小心信号和竞争条件：

存在的风险：



特权进程应该避免之前成立但是现在不成立的条件操作。

1. 执行文件操作和文件IO的缺陷：
2. 使用umask设置。
3. 确定创建文件的用户身份。
4. 还有别的具体查看书本。
5. 不要完全相信输入和环境：

不要相信环境列表，尤其是PATH和IFS。

对用户输入要进行防御。

避免对运行时环境进行可靠性假设。

1. 小心缓冲区的溢出：

使用一些限定输入大小的函数。

1. 小心拒绝服务攻击：
2. 检查返回的状态和安全处理失败的情形。